TLC 1.3 Technical Installation Notes

Intended Audience

These notes are intended to be read by a technical person familiar with Java, Grails, Tomcat (or equivalent), SQL databases and your operating system. This person is referred to within the application as the System Administrator. The special System Administration options are only available to the System Administrator who, through improper use of these facilities, can make the application inoperable. Obviously, therefore, the System Administrator must be a person, or persons, of considerable technical competence.

Source Code Download

The downloaded zip file is named tlc-x.y.zip where x and y are the major and minor release numbers. The contents of the zip file are a Grails application in a directory called tlc. Be careful when unzipping the file that you do not overwrite any existing tlc directory contents.

Minimum System Requirements

1. Java version 1.6 SE
2. 1GB of memory available to Java
3. Grails version 1.3.1
4. A servlet container such as Tomcat
5. Optionally, a web server such as Apache
6. MySQL database server

Notes:

If you use Java from Sun Microsystems Inc, then you should split the available memory to allow 256MB for PermGen space and 768MB to the heap with start up parameters such as: -XX:MaxPermSize=256M -Xmx768M. You may also wish to set the initial heap size for faster start-up times. In production mode, this probably means editing the configuration files of your servlet container. In development mode, when you first want to try the system out, you can either set your JAVA_OPTS in your environment or edit the $GRAILS_HOME/bin/startGrails shell script (or .bat batch file) changing the standard JAVA_OPTS line to that required.

Although Grails uses Hibernate which should therefore allow the use of databases other than MySQL, no testing has yet been done with databases other than MySQL.

Libraries

The tlc/lib directory contains the jar libraries that the application uses. Your servlet container installation may already have libraries such as activation and mail available at container level, in which case these libraries could be removed from tlc/lib, although is does no harm to leave them there.

A MySQL database connector jar file is included in the tlc/lib directory. You may replace or remove this file to suit your needs.

Keep the org.springframework.test.jar library in the tlc/lib directory since it is used in both the production and development environments, not just the test environment. Keep it up to date if you upgrade Grails. It will be in the grails-n.n.n/lib directory.

The jasperreports-n.n.n.jar in the tlc/lib directory contains one modified file and two new files above and beyond the contents of a normal JasperReports jar file. If you upgrade to a newer version of the JasperReports runtime library, extract the three files CODE2000.TTF, fonts1259239518045.xml and jasperreports_extension.properties files from the existing jar file and add them to the jar file of the new JasperReports distribution. You may also wish to check that the application's poi and iText jar files in tlc/lib are consistent with those of the new version of JasperReports.

Configuration

Update the contents of the DataSource.groovy settings for your database and check that the database connection pooling settings are as you require them. In particular, check that the validationQuery string is a valid SQL query statement for your database.

Check through all the settings in Config.groovy. There are a number of settings specific to the TLC application.

The application comes as standard to use Ehcache as Hibernate's cache provider. Ehcache is set to use only memory caching and not disk caching. If you wish to change this then you can edit the ehcache.xml file in the tlc/grails-app/conf directory to set the caching up as you wish.

Running The Application

TLC is a relatively large application and you would be best advised not to use the Grails in-memory database even for development or testing purposes. You should connect to a 'real' database.

Your server should ideally have a connection to the Internet if you intend to take advantage of the automatic update of foreign currency exchange rates from Yahoo! that the application provides.

Before running or deploying the application, ensure your browser has JavaScript enabled and, as far as possible, has its locale (language and country) set correctly.

Allow around five to ten minutes (depending on the speed of your system) for TLC to fire up since it has a lot of tables to create and initial data to load. Once the server is running, log in as the system administrator using an id of system and password of sysadmin. You should now change the login id and password of the system administrator and, optionally, create a secondary system administrator 'just in case'.

After a couple of minutes (there is an intial start-up delay built in), have a look at the Task Queue to

see that the initial tasks executed without error. This is a good indicator that all is well with your installation. Have an in-depth look around the System Administration menu options – but don't actually do anything – if this is the first time you have seen them.

The application can run in one of two modes: 'demo' and 'live'. You can see the application running in demo mode at the Wholly Grails web site. Demo mode allows people to set up their own companies and 'play around' with the system and also 'cleans up' after them. This can be very useful for a development installation and as a Testing or Training installation for end users. By default, however, a new system will boot up for the first time in 'live' mode. If you wish this to be a demo system, change the isDemoSystem System Setting from false to true.

Post Installation

Unless you have changed it, the DataSource.groovy configuration file will have created the database tables in 'update' mode. This is very convenient when ongoing changes are made to the structure of the database tables, but has one drawback on initial installation... it is unlikely to have created certain additional indexes needed by the application. Consequently, you need to check for the existence of the following table indexes and (as is probable), create them if they are not there using SQL statements in your database's control panel such as the following (which are in MySQL dialect):

```
create index doc_created_idx on document (type_id, date_created)
create index reconciliation_idx on general_transaction (reconciliation_key, reconciled)
create index waiting_idx on queued_task (current_status, preferred_start)
create index next_run_idx on task (next_scheduled_run)
create index sysmenu_parent_idx on system_menu (parent)
```

You might also want to 'update statistics' after creating these indexes since there is already a significant amount of data in some of the tables.

If you enabled the database maintenance options in Config.groovy, then you should use the Task Definition facility in the Company Administration option of the System company to give the 'maintain' task a 'Next Scheduled Run' date and time.

The application includes an 'automatic payment of suppliers' facility which, if you intend to use it, requires an additional Groovy program writing to actually pass payment information from the company to your bank(s). A skeleton version of such a program is included in the tlc directory in a file called SAMPLE_PAYMENT_SERVICE.

The System Company

When first installed, only one company is created by the application. This is a company called System and is very special. It is used to hold 'template' data for other 'real' companies that you will create. Consequently, be very careful that you never change any data belonging to the System company unless you are absolutely certain you know what you are doing.

If you change anything about the System company without a full understanding of the implications,

then all bets are off!

Hand Over

The remaining instructions are for 'live' systems as opposed to 'demo' systems.

Create the company that you intend to run on the system – remember that company System is not a 'real' company. Ensure the country, language and currency settings are correct for the new company.

Create a new user who will be the Company Administrator for the new company (your accountant?). Fill in the login id, name, email, password and confirmation, security question and answer, country and language fields. Do not alter the other fields.

Give the new user access to the new company by creating a new company user for them, selecting the new company as the company to add them to.

Assign the Company Administrator role to the new user in the new company.

You can now log out and hand the system over to the new Company Administrator for use.

